# CS 595: Assignment #7

Due on Sunday, November 9, 2014

*Dr Nelson 4:20PM*

**VICTOR NWALA**

# Contents

# Problem 1

1. Using D3, create a graph of the Karate club before and after the split.
- Weight the edges with the data from: http://vlado.fmf.uni-lj.si/pub/networks/data/ucinet/zachary.dat
- Have the transition from before/after the split occur on a mouse click.

To answer this question I downloaded a template from D3 to suite my graph. I also download and modified the json version of the karate club graph.
To implement the on mouse click function, I consulted Alexander Nwala and he gave a few ideas which I used. I reflected the weight of the links as a function of the size of the width of the link. The more the weight the bigger the width of the link. The graph can be viewed on my server here:
`http://www.cs.odu.edu/~vnwala/cs595/graph.html`
Listing 1 shows a Html script.

Listing 1: Html Script FOR D3 Visualization for Karate club graph

```
<!DOCTYPE html>
<head>
<h2 align="center" > KARATE CLUB GRAPH SHOWING TWO DISTICT GROUP</h2>
<h3 align="center"> click any node to see groups with distinct colours</h3>
</head>
<meta charset="utf-8">
<style>

.node {
  stroke: #fff;
  stroke-width: 1.5px;
}

.link {
  stroke: #999;
  stroke-opacity: .6;
}

.text {
  fill: #000;
  font: 10px sans-serif;
  pointer-events: none;
}

</style>
<body>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>


var Flag = 1;


var width = 960,
    height = 500;
```

Problem 1 continued on next page...

```
var color = d3.scale.category10();


var force = d3.layout.force()
    .charge(-120)
    .linkDistance(120)
    .size([width, height]);

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height);

d3.json("graph.json", function(error, graph) {
  force
      .nodes(graph.nodes)
      .links(graph.links)
      .start();

  var link = svg.selectAll(".link")
      .data(graph.links)
    .enter().append("line")
      .attr("class", "link")
      .style("stroke-width", function(d) { return Math.sqrt(d.weight*4); });

  var node = svg.selectAll(".node")
      .data(graph.nodes)
    .enter().append("circle")
      .attr("class", "node")
      .attr("r", 5)
       .on("click", Click)
      .style("fill", function click(d) { return color(d.color); })
      .call(force.drag);




  node.append("title")
      .text(function(d) { return d.name; });









  force.on("tick", function() {
    link.attr("x1", function(d) { return d.source.x; })
        .attr("y1", function(d) { return d.source.y; })
        .attr("x2", function(d) { return d.target.x; })
        .attr("y2", function(d) { return d.target.y; });

    node.attr("cx", function(d) { return d.x; })
```

```
        .attr("cy", function(d) { return d.y; });
  });
});


function Click(d)
    {

      if(Flag==1)
      {

        d3.selectAll('.node').style('fill', function(d) { return color(d.club); });

        Flag = 0;
      }
      else
      {

        d3.selectAll('.node').style('fill', function(d) { return color(d.color); });


        Flag = 1;
      }
    }
</script>
```

Figure 1: Graph on server before split

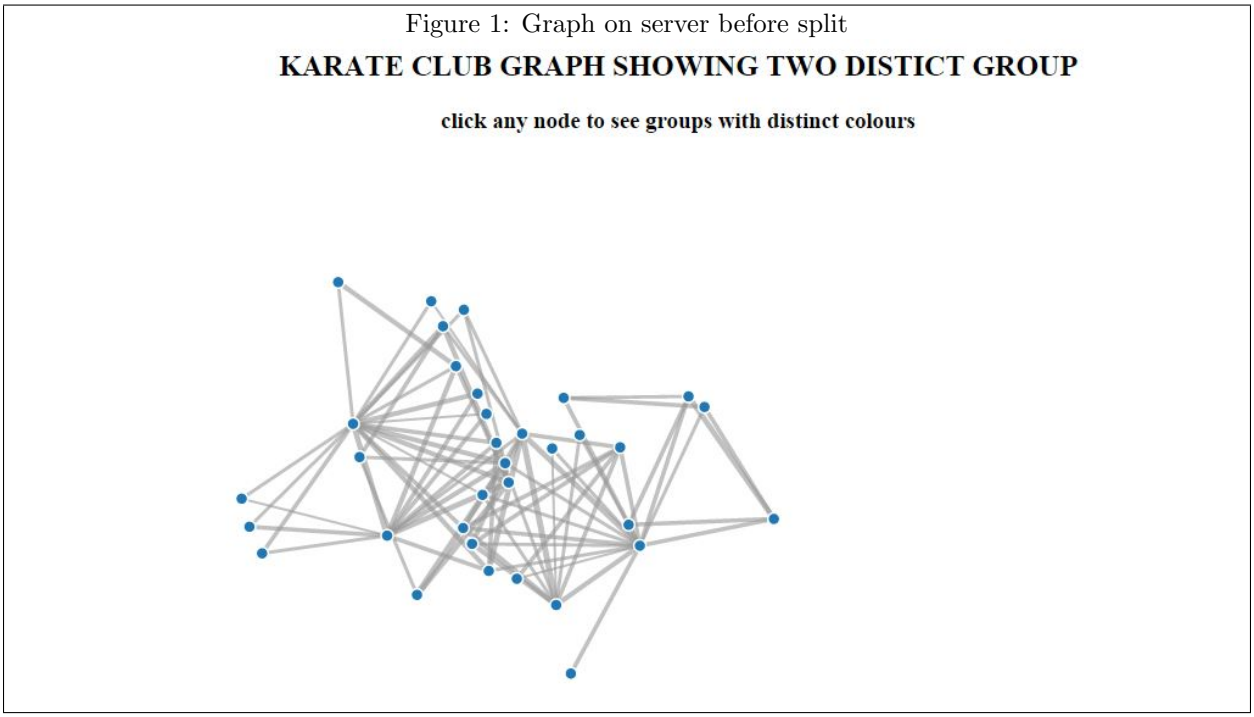
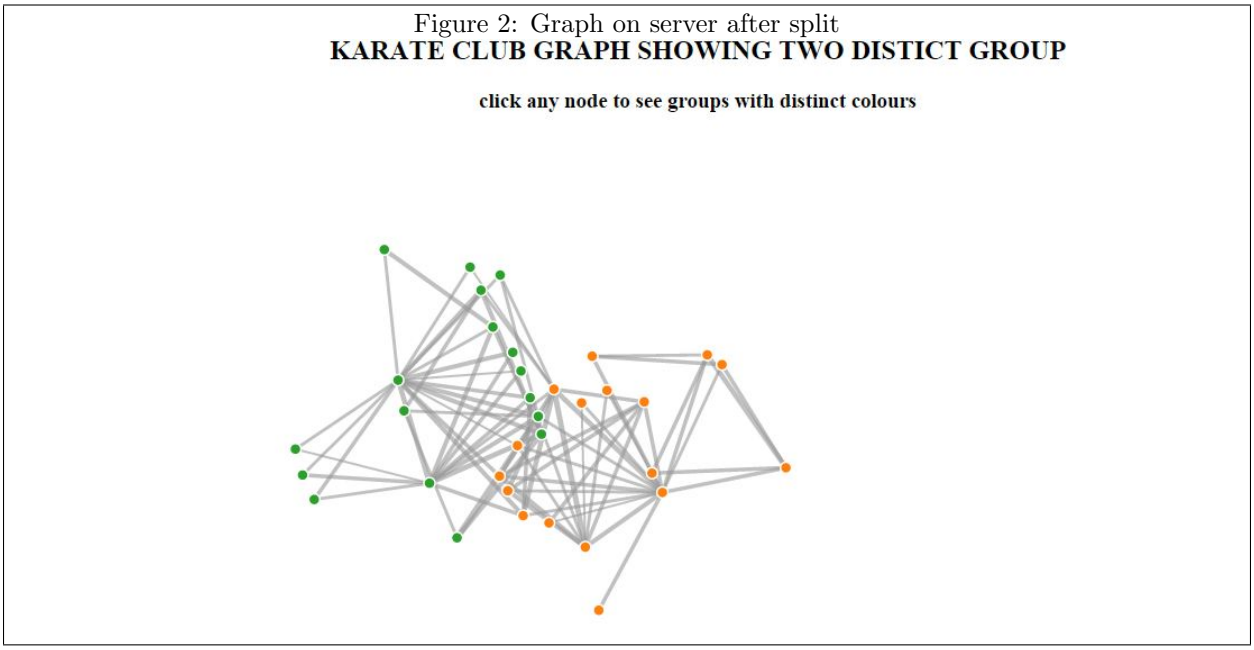


Figure 2: Graph on server after split



# References

[1] mbostock. Force-directed graph. http://bl.ocks.org/mbostock/4062045, 12 November 2012.

[2] Cyrille Rossant. Visualizing a networkx graph in the ipython notebook with d3.js. 2014.

[]